Speeding up Glauber Dynamics for Random Generation of Independent Sets

Rémi Varloot* Ana Bušić† Anne Bouillard‡

April 20, 2015

Abstract

The maximum independent set (MIS) problem is a well-studied combinatorial optimization problem that naturally arises in many applications, such as wireless communication, information theory and statistical mechanics.

MIS problem is NP-hard, thus many results in the literature focus on fast generation of maximal independent sets of high cardinality. One possibility is to combine Gibbs sampling with coupling from the past arguments to detect convergence to the stationary regime. This results in a sampling procedure with time complexity that depends on the mixing time of the Glauber dynamics Markov chain.

We propose an adaptive method for random event generation in the Glauber dynamics that considers only the events that are effective in the coupling from the past scheme, accelerating the convergence time of the Gibbs sampling algorithm.

1 Introduction

An *independent set* of a graph is a set such that no two nodes in the subset are connected by an edge. The maximum independent set (MIS) problem is to find the set of mutually nonadjacent nodes with the largest cardinality. MIS is a well-studied combinatorial optimization problem that naturally arises in many applications, such as statistical physics (where it is known as the hard core gas model) [7, 4], information theory [3] and wireless communication [14, 8].

Generating independent sets is one of the key building blocks of the wireless CSMA [5, 15, 14, 8]. The interference in a wireless network can be modeled by a conflict graph. The nodes are the links and there is an edge between two nodes if the corresponding communication links cannot transmit simultaneously. At each step of the protocol a set of the communication links is chosen that forms an independent set in this conflict graph. In queue-based CSMA, the nodes have weights that are the queue sizes at the wireless links. Ideally, one should compute a maximum weight independent set (MWIS). However, MWIS problem is NP-hard and hard to approximate in general [16]. Many papers focus on finding *good enough* independent sets, see [14] for a more detailed discussion in the context of CSMA. In [13] the authors consider a message passing approximation algorithm for MWIS problem. They show that, if initialized using uninformative messages, their algorithm returns an optimal value if it converges. However, the convergence is proven only for the case of a bipartite graph with a unique MWIS.

The focus of this paper is on the approximations to the MWIS problem using Glauber dynamics [10] over the space of independent sets of the interference graph. To simplify exposition, much of the analysis is focused on the special case in which all the weights are equal; extensions to the

*Microsoft Research - Inria Joint Centre, France.

†Inria and the Computer Science Dept. of École Normale Supérieure, Paris, France.

‡Computer Science Dept. of École Normale Supérieure, Paris, France.

The work presented in this paper has been carried out at LINCS (www.lincs.fr) and is supported by the French National Research Agency grant ANR-12-MONU-0019.

completely general case is explained In Section 4 (at the end of Subsection 4.1.). Glauber dynamics are defined by a (reversible) Markov process that has as a stationary distribution a Gibbs measure

$$\pi(A) = \frac{\lambda^{card(A)}}{Z_\lambda},$$

where A is an independent set of the graph, λ is a parameter called fugacity, and Z_λ is the normalization constant. For $\lambda = 1$, this corresponds to the uniform distribution over all independent sets and when λ goes to infinity, the Gibbs measure is concentrated on the maximum independent sets. For high values of λ, the mixing time of this dynamics becomes prohibitively large. Furthermore, the existing bounds for the mixing time of graphs are limited to the bounded degree case [7, 8, 15].

We combine the Glauber dynamics with the coupling from the past (CFTP) construction for stationarity detection of Markovian dynamics. CFTP is an exact simulation technique introduced by Propp and Wilson [12]. The original algorithm is computationally efficient only under some monotonicity assumptions on the Markovian dynamics. In the general case, the CFTP algorithm requires the construction of one trajectory for each initial condition, which is computationally intractable in most applications. Huber [7] proposed a more general CFTP algorithm that is based on a construction of a bounding chain that avoids this dependence on the cardinality of the state space. However, this comes with a new penalty - the running time of the algorithm can be much larger than the mixing time of the original Markovian dynamics. Intuitively, many transitions have no effect on the bounding chain, inducing useless steps in the CFTP algorithm.

The main contribution of this paper is a new CFTP algorithm that uses an adaptive event table and avoids generation of events that do not have any effect on the bounding chain. The idea of *skipping* passive events is very natural, but it is far from straightforward to see how this can be combined with the CFTP scheme without introducing a bias. The proof that our algorithm terminates in finite expected time and provides an exact sample from the stationary distribution of the Markovian dynamics is stated as our main result in Theorem 2.

We illustrate the speed-up of Glauber dynamics for the independent sets on a toy example of a star network, for which we can derive bounds for the computation time of our algorithm. We show that, unlike the initial Glauber dynamics, our algorithm does not depend on λ. Similar results are obtained numerically in Section 4.3 for the Barabási-Albert model [1]. We also compare the proposed algorithm against Dyer and Greenhill dynamics [2] that use a *swap* operation to speed up the convergence.

The paper is organized as follows. An overview of the coupling from the past construction for the exact sampling from the stationary distribution of a Markov chain is given in Section 2. Our main contribution is presented in Section 3. We start in Section 3.1 by introducing the idea of active and passive events and the construction of a dynamic event table that contains only active events. Section 3.2 contains a detailed discussion on the skipping of events in the CFTP scheme, summarized in Algorithm 3. The validity of the algorithm is provided by Theorem 2. Section 4 contains the application to independent sets, while Section 5 concludes the paper.

2 Coupling From the Past

Throughout this section, $\mathcal{S}$ designates a finite state space.

We study some properties of ergodic Markov chains over $\mathcal{S}$, namely in the case of a joint distribution between multiple Markov chains.

2.1 Mixing Time

The mixing time is an indicator of how long it takes a Markov chain to forget its initial distribution. In essence, it measures how long Markov chain Monte Carlo methods must run before being "close" to the stationary distribution.

Let ρ and π be two probability distributions over $\mathcal{S}$. Recall that the *total variation distance* $\|\cdot\|_{\mathrm{TV}}$ between the two is defined as

$$\|\rho - \pi\|_{\mathrm{TV}} = \max_{A \subseteq \mathcal{S}} |\rho(A) - \pi(A)|.$$

Definition 1 (Mixing time)**.** *Let M be the transition matrix of an ergodic Markov chain over $\mathcal{S}$, with stationary distribution π. For all $x \in \mathcal{S}$, let $(X_i(x))_{i \in \mathbb{N}}$ be the Markov chain with transition matrix M and initial state x, and, for all $i \in \mathbb{N}$, call ρ_i^x the distribution of $X_i(x)$. The* mixing time t_{mix} *of M is defined as*

$$t_{\mathrm{mix}} = \min \left\{ i \in \mathbb{N} \;\middle|\; \max_{x \in \mathcal{S}} \|\rho_i^x - \pi\|_{\mathrm{TV}} \leq \frac{1}{4} \right\}.$$

Notice that the mixing time is defined for transition matrices rather than Markov chains, as its definition considers *all* Markov chains generated by a given transition matrix.

Property 1. *Using the above notation, if, at instant $i \in \mathbb{N}$, there exists an event A and a state $x \in \mathcal{S}$ such that*

$$|\rho_i^x(A) - \pi(A)| \geq \frac{1}{4},$$

then $i \leq t_{\mathrm{mix}}$.

This last property serves to derive lower-bounds on the mixing time of certain transition matrices.

2.2 Coupling Time

Just like the mixing time allows us to measure how quickly Markov chains converge towards their stationary distributions, the coupling time measures how long it takes before two or more Markov chains "meet" in a same state.

Definition 2 (Coupling)**.** *Let K be a finite set of indices. For all $k \in K$, let*

$$X^k = \left(X_i^k\right)_{i \in \mathbb{N}}$$

be a Markov chain over $\mathcal{S}$. A coupling *of the X^k is a family of joint Markov chains*

$$\mathcal{X} = \left(\left(\mathcal{X}_i^k\right)_{i \in \mathbb{N}} \right)_{k \in K}$$

defined over a same probability space such that, for all $k \in K$, the marginal distribution of $\mathcal{X}^k = \left(\mathcal{X}_i^k\right)_{i \in \mathbb{N}}$ is that of X^k.

Definition 3 (Coupling Time)**.** *Let X be a coupling of Markov chains over $\mathcal{S}$. We say X has* coupled *at instant i if all the X_i^k, $k \in K$ are equal.*

We furthermore define the coupling time τ *of X as the first instant at which X has coupled, i.e.*

$$\tau = \min \left\{ i \in \mathbb{N} \;\middle|\; \forall (k,l) \in K^2, X_i^k = X_i^l \right\},$$

with the convention that $\min \emptyset = +\infty$.

Note that, unlike the mixing time, the coupling time is a random variable.

2.3 Markov Automata

Markov automata are a convenient means of writing random mapping representations [9] when defining a coupling between Markov chains that have the same transition matrix.

Definition 4 (Markov automaton)**.** *A* Markov automaton *is a quadruple $\mathcal{A} = (\mathcal{S}, A, D, \cdot)$, where:*

- *$\mathcal{S}$ is a finite state space;*
- *A is a finite alphabet;*
- *D is a probability distribution over A;*
- *$\cdot$ is an action by the letters of A on the states of $\mathcal{S}$:*

$$\cdot : \begin{cases} \mathcal{S} \times A & \to \mathcal{S} \\ (x, a) & \mapsto x \cdot a \end{cases}$$

We make the assumption that, for all $a \in A$, $D(a) > 0$. If not, it is possible to build a reduced Markov automaton satisfying this property by removing all the letters $a \in A$ such that $D(a) = 0$.

Let $A^* = \bigcup_{k \in \mathbb{N}} A^k$ be the set of finite words, $A^\omega = A^{\otimes \mathbb{N}}$ be the set of infinite words, and $A^\infty = A^\star \cup A^\omega$.

For a *word* $u \in A^\infty$ and for $-\infty \leq i, j \leq \infty$, we denote $u_{i \to j}$ the subword $(u_i, \ldots, u_j)$ if $i \leq j$, or ϵ if $j < i$.

For convenience, we furthermore write $S \cdot a$ for $\{x \cdot a \mid x \in S\}$ and $x \cdot u_{1 \to n}$ for $x \cdot u_1 \cdot \ldots \cdot u_n$, such that $S \cdot u_{1 \to n}$ stands for

$$\{x \cdot u_1 \cdot \ldots \cdot u_n \mid x \in S\}.$$

Let $\mathcal{A} = (\mathcal{S}, A, D, \cdot)$ be a Markov automaton, and $u_{1 \to \infty} \sim D^{\otimes \mathbb{N}}$. For all $x \in \mathcal{S}$, define $X(x) = (X_i(x))_{i \in \mathbb{N}}$ by

$$\forall i \in \mathbb{N}, X_i(x) = x \cdot u_{1 \to i},$$

i.e. $X_i(x)$ is the state reached when starting in x and reading $u_{1 \to i}$.

Property 2. *For every $x \in \mathcal{S}$, $X(x)$ is a Markov chain, called the Markov chain* generated *by $\mathcal{A}$ and x. Furthermore, these Markov chains have the same transition matrix $M_\mathcal{A}$.*

The family $X = (X(x))_{x \in \mathcal{S}}$ is a natural coupling between these Markov chains, called the *grand coupling* of $\mathcal{A}$.

If there exists a word $u_{1 \to n}$ such that $\mathcal{S} \cdot u_{1 \to n}$ is a singleton, we say that $\mathcal{A}$ *couples*, and call $u_{1 \to n}$ a coupling word.

Property 3 ([12])**.** *If $\mathcal{A}$ couples, we have that $M_\mathcal{A}$ is ergodic, and that*

$$\mathbf{P}\left\{\lim_{i \to \infty} |X_i| = 1\right\} = 1,$$

i.e. X a.s. has a finite coupling time.

The reciprocal is not true: it is possible to construct a non-coupling Markov automaton $\mathcal{A}$ such that $M_\mathcal{A}$ is ergodic.

If $\mathcal{A}$ couples, we define its stationary distribution and mixing time as those of $M_\mathcal{A}$, and its coupling time τ as that of X. The coupling time of a Markov automaton is closely linked to its mixing time, as shown in the following property.

Property 4 ([9])**.** *If $\mathcal{A}$ couples, the expected coupling time of $\mathcal{A}$ is lower-bounded by the mixing time t_{mix} of $M_\mathcal{A}$, i.e.*

$$t_{\text{mix}} \leq \mathbf{E}[\tau].$$

It is important to underline that the distribution of the unique value of the grand coupling at the first moment of coupling is *not* distributed according to the stationary distribution of $\mathcal{A}$ [6]. We now introduce an algorithm which uses the grand coupling of a Markov automaton to obtain that distribution.

2.4 Coupling from the Past

Let $\mathcal{A} = (\mathcal{S}, A, D, \cdot)$ be a coupling Markov automaton and $u_{-\infty\to-1} \sim D^{\otimes\mathbb{N}}$. Define $(S_i)_{i\in\mathbb{N}}$ by

$$\forall i \in \mathbb{N}, S_i = \mathcal{S} \cdot u_{-i\to-1}$$

and let τ^b be the first i for which S_i is a singleton. τ_b is called the backwards coupling time of the Markov automaton.

Theorem 1 ([12])**.** *Using the above notations, we have that the unique element of S_{τ^b} is a.s. distributed according to the stationary distribution of $\mathcal{A}$, and that $\mathbf{E}\left[\tau^b\right] = \mathbf{E}\left[\tau\right]$, where τ is the coupling time of $\mathcal{A}$.*

This method for generating random variables according to the stationary distribution of a Markov chain is called *coupling from the past* (CFTP). The sequence $u_{-\infty\to-1}$ is called the *generating sequence*. The corresponding algorithm is given in Algorithm 1.

Algorithm 1 Coupling From the Past (CFTP)

```
function CFTP((S, A, D, ·))
    for s ∈ S do
        S(s) ← s
    end for
    repeat
        a ← Draw(D)
        for s ∈ S do
            T(s) ← S(s · a)
        end for
        S ← T
    until |S(S)| = 1
    return UniqueElementOf(S(S))
end function
```

Property 5. *The expected complexity of the CFTP algorithm is $O\left(|\mathcal{S}|\,\tau\gamma\right)$, where γ is the computation time of $\cdot$.*

The complexity is linear in $|\mathcal{S}|$, which can be very large. A workaround for this is to use *bounding chains* [7].

Definition 5. *Let $\mathcal{B}$ be a subset of the power set of $\mathcal{S}$, containing $\mathcal{S}$, and let $\circ : \mathcal{B} \times A \to \mathcal{B}$ be an operator such that for all $a \in A$ and $B \in \mathcal{B}$,*

$$x \in B \Rightarrow x \cdot a \in B \circ a.$$

The bounding chain *of $\mathcal{S} \cdot u_{1\to n}$ induced by $(\mathcal{B}, \circ)$ is the sequence $\mathcal{S} \circ u_{1\to n} = \mathcal{S} \circ u_1 \circ \ldots \circ u_n$.*

Notice that, for any $n \in \mathbb{N}$ and $u_{1\to n} \in A^n$,

$$\mathcal{S} \cdot u_{1\to n} \subseteq \mathcal{S} \circ u_{1\to n},$$

hence the term *bounding chain*.

Let $u_{-\infty\to-1} \sim D^{\otimes\mathbb{N}}$. For all $i \in \mathbb{N}$, we have that

$$\mathcal{S} \cdot u_{-i\to-1} \subseteq \mathcal{S} \circ u_{-i\to-1}.$$

As a consequence, if there exists a word $u_{-n\to-1}$ such that $\mathcal{S} \circ u_{-n\to-1}$ is a singleton, then so is $\mathcal{S} \cdot u_{-n\to-1}$, and they contain the same element.

From this, we derive a variant of the CFTP algorithm in which we iteratively compute

$$B_i = \mathcal{S} \circ u_{-i \to -1}$$

until we obtain a singleton. The backwards coupling time τ^b of the bounding chain is then defined as the hitting time of the set of singletons. Note that the sequence

$$\left(\mathcal{S} \circ u_{-i \to -j}\right)_{j \in [\![-i, -1]\!]}$$

must now be recomputed at each iteration. This yields an overall complexity in $O\left(\tau^2 \Gamma\right)$, where Γ represents the computation time of $\circ$, often small compared to $\gamma\,|\mathcal{S}|$.

The appearance of a quadratic dependency in τ can be overcome by doubling the period at each iteration [12]. The algorithm is given in Algorithm 2, and has a complexity of

$$O\left(\tau \Gamma\right).$$

Algorithm 2 CFTP with Bounding Chains

```
function BOUNDED-CFTP((S, A, D, ·))
    w ← ϵ
    k ← 1
    repeat
        w ← DRAW(D^{⊗k}) · w
        B ← S ∘ w
        k ← 2k
    until |B| = 1
    return UNIQUEELEMENTOF(B)
end function
```

3 Skipping

Skipping was introduced in [11], in a form close to what we call here *incremental skipping*, as a means of speeding up the CFTP algorithm by avoiding certain "passive" events. It is introduced here alongside our own approach, *oracle skipping*.

We first introduce these two methods on forward coupling chains, and show their computational similarities. We then adapt oracle skipping to the CFTP algorithm, and give proof of its correctness.

3.1 Oracle and Incremental Skipping

Consider a forward coupling algorithm, with bounding chain $B = (B_i)_{i \in \mathbb{N}}$, and let $u_{1 \to \infty}$ be the corresponding sequence of letters:

$$\forall i \in \mathbb{N}, B_i = \mathcal{S} \circ u_{1 \to i}.$$

We say a letter a is *inactive at instant* i if $B_i \circ a = B_i$, and that it is *active* otherwise. Let $\mathfrak{A}_{u_{1 \to i-1}}$ be the set of active events at instant i. With these definitions, we consider a new bounding chain $B^{\mathcal{O}} = \left(B_i^{\mathcal{O}}\right)_{i \in \mathbb{N}}$ such that

$$\forall i \in \mathbb{N}, B_i^{\mathcal{O}} = \mathcal{S} \circ v_{1 \to i},$$

with the $v_{1 \to \infty}$ drawn according to D *conditioned to being active letters*:

$$\forall i \in \mathbb{N}, v_i \sim D\left(\cdot \,\middle|\, \mathfrak{A}_{v_{1 \to i-1}}\right).$$

This method of generating a bounding chain is called *oracle skipping*.

Despite oracle skipping being easy to manipulate on a theoretical level, it can be difficult to get an efficient implementation of the algorithm. This is due to the cost of computing the conditional distribution at each iteration.

The original skipping algorithm [11], *incremental skipping*, provides a workaround for this: rather than recomputing the entire distribution of events at each step, the algorithm updates its distribution incrementally.

Consider that the resulting bounding chain $B^{\mathcal{I}} = \left(B_i^{\mathcal{I}}\right)_{i\in\mathbb{N}}$ is obtained from a word $v'_{1\to n}$, such that

$$\forall i \in \mathbb{N}, B_i^{\mathcal{I}} = \mathcal{S} \circ v'_{1\to i}.$$

Begin by setting $D^0 = D$. For all $i \in \mathbb{N}$, v'_i is drawn according to D^i. If $v'_i \in \mathfrak{A}_{v'_{1\to i-1}}$, set $D^{i+1} = D$, otherwise define D^{i+1} by removing v'_i from the set of possible events:

$$\forall a \in A, D^{i+1}(a) = D^i\left(a \mid a \neq v'_i\right).$$

This process constructs the distributions $\left(D^i\right)_{i\in\mathbb{N}}$ recursively by removing at most one event at each iteration.

Though incremental skipping is more efficient in the case of complicated conditional distributions, the complexity of the rest of the algorithm is greater than in the case of oracle skipping. Furthermore, it is very difficult to obtain theoretical bounds with incremental skipping.

The following property justifies studying oracle skipping to derive bounds of coupling time of skipping algorithms, regardless of implementation.

Property 6. *Call $\tau_{\mathcal{O}}$ and $\tau_{\mathcal{I}}$ the coupling times of $B^{\mathcal{O}}$ and $B^{\mathcal{I}}$. We have that*

$$\tau_{\mathcal{O}} \leq \tau_{\mathcal{I}} \leq M \cdot \tau_{\mathcal{O}},$$

where $M = |A|$ is the cardinality of the event set.

Proof. Notice that $v_{1\to\infty}$ can be obtained from $u_{1\to\infty}$ by removing *all* of its passive letters, and $v'_{1\to\infty}$ can be obtained from $u_{1\to\infty}$ by removing *some* of its passive letters. Doing so results in a coupling of the three bounding chains $B^{\mathcal{O}}$, $B^{\mathcal{I}}$ and B such that there exists increasing functions ϕ and ψ from $\mathbb{N}$ to $\mathbb{N}$ satisfying:

$$B_i^{\mathcal{O}} = B_{\phi(i)}^{\mathcal{I}} = B_{\psi(\phi(i))}.$$

The first inequality is a direct consequence of this, since $v_{1\to\tau_{\mathcal{O}}}$ is therefore a subsequence of $v'_{1\to\tau_{\mathcal{I}}}$.

For the second inequality, notice that two active events in $v'_{1\to\tau_{\mathcal{I}}}$ are separated by a sequence of pairwise different passive events, and the distance between the two is therefore at most M. This implies that $v_{1\to\tau_{\mathcal{O}}}$ contains at least one letter out of M from $v'_{1\to\tau_{\mathcal{I}}}$, i.e.

$$\tau_{\mathcal{O}} \geq \frac{\tau_{\mathcal{I}}}{M}.$$

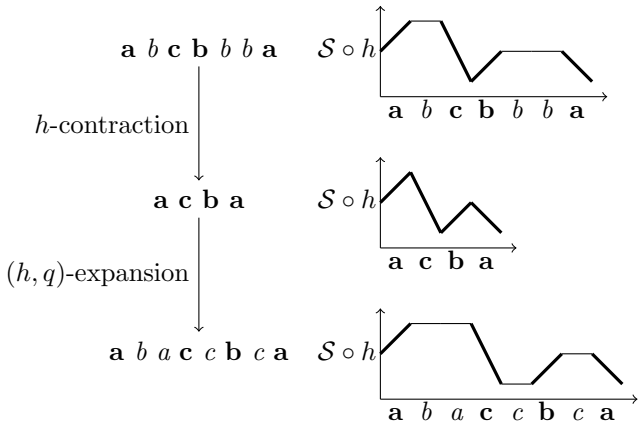
This concludes the proof. □

3.2 CFTP with Oracle Skipping

We now adapt oracle skipping to the CFTP algorithm. For an adaptation of incremental skipping, see [11].

The difficulty in implementing a CFTP algorithm with oracle skipping lies in the fact that, as we move backwards in time, the state of the system at a *fixed* instant $-k$ changes. The event u_{-k} can therefore start out as active, then become passive, then active again, etc. each time we go further back in time. Whereas removing events that have become passive is not difficult, a passive event that was removed and that ought to be active once more cannot simply be pushed back in; keeping the event in memory would imply drawing every event, which defeats the purpose.

The solution adopted here consists in dropping passive letters completely, and inserting active letters according to an adequate distribution, one that preserves the dynamics of the initial bounding chain. We give the details of this algorithm, and prove its correctness.



Active letters are shown in bold font.

Figure 1: Contracting and Expanding

To begin, we introduce a delimiter, denoted $\sharp$, used to split up our sequence of events, and two new operations: contraction, which consists in removing passive letters, and expansion, through which these passive letters are added back into a contracted word.

Fix a Markov automaton $\mathcal{A} = (\mathcal{S}, A, D, \cdot)$. Let $\mathcal{B}$ be a subset of the power set of $\mathcal{S}$ and $\circ$ be an operator, such that $(\mathcal{B}, \circ)$ induces a bounding chain for the grand coupling of $\mathcal{A}$. Define the delimiter $\sharp$ as a letter that leaves states unchanged:

$$\forall x \in S, x \cdot \sharp = x \quad \text{and} \quad \forall B \in \mathcal{B}, B \circ \sharp = B.$$

Let $A_\sharp = A \cup \{\sharp\}$. For all $q \in [0,1]$, call D_q be the distribution over $A_\sharp$ such that:

$$D_q(\sharp) = q \quad \text{and} \quad \forall a \in A, D_q(a) = (1-q) \cdot D(a).$$

Furthermore, for any subset S of $\mathcal{S}$, let

$$\mathfrak{A}(S) = \{a \in A \mid S \neq S \circ a\} \cup \{\sharp\}$$

be the set of *active* letters and

$$\mathfrak{P}(S) = \{a \in A \mid S = S \circ a\}$$

be the set of *passive* letters. To simplify notations, we write

$$\mathfrak{A}_{a_{1\to k}} = \mathfrak{A}(\mathcal{S} \circ a_{1\to k}) \quad \text{and} \quad \mathfrak{P}_{a_{1\to k}} = \mathfrak{P}(\mathcal{S} \circ a_{1\to k}),$$

as these are the active and passive letters of the bounding chain after having read $a_{1\to k}$.

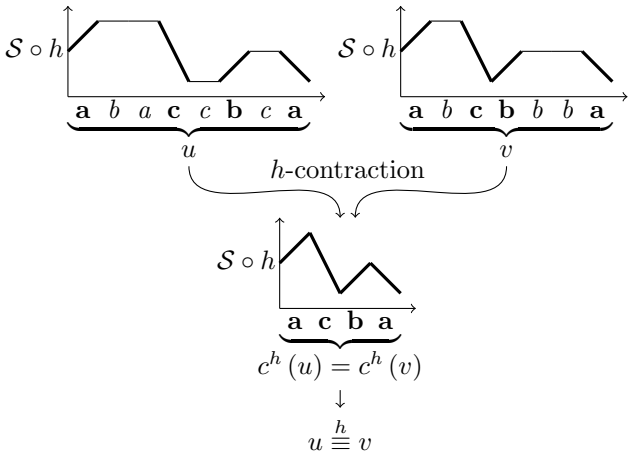
Notice that $\sharp$ is an exception: it never modifies the state of the chain, yet is considered to be an active letter nonetheless.

We now define contraction and expansion. These are illustrated in Figure 1.

Definition 6 (h-contracting)**.** *Contraction consists in removing the passive letters in a word. For a given history $h \in A_\sharp^*$, call* h-contraction *the operation $c^h : A^\infty \to A^\infty$ defined recursively by $c^h(\epsilon) = \epsilon$ and*

$$c^h(a \cdot u) = \begin{cases} a \cdot c^{h \cdot a}(u) & \text{if } a \in \mathfrak{A}_h \\ c^h(u) & \text{otherwise.} \end{cases}$$

Notice that contraction is idempotent. A word invariant under c^h is called a *h-contracted* word.



Active letters are shown in bold font.

Figure 2: h-equivalence

Definition 7 ((h,q)-expanding)**.** *Expansion consists in inserting passive letters in a word. For a given history $h \in A_\sharp^*$ and $q \in [0,1]$, call* (h,q)-expansion *the operation $e_q^h : A^\infty \to A^\infty$ defined recursively by $e_q^h(\epsilon) = \epsilon$ and*

$$e_q^h(a \cdot u) = \begin{cases} p \cdot e_q^h(a \cdot u) & \text{with probability } D_q(\mathfrak{P}_h) \\ a \cdot e_q^h(u) & \text{with probability } D_q(\mathfrak{A}_h), \end{cases}$$

where p is a passive letter drawn independently according to

$$D_q(\cdot \mid \mathfrak{P}_h),$$

the distribution D_q restricted to inactive letters.

Note that, during expansion, the number of passive letters inserted before each letter in the initial word is geometrically distributed.

Applying e_q^h to a contracted word corresponds to constructing what the word *"could have been"* before it was contracted by c^h, under the assumption that its letters were originally i.i.d. according to D_q and that it ended with an active letter.

Definition 8 (h-equivalence)**.** *Let $h \in A_\sharp^*$ be a history, and $u \in A_\sharp^\infty$ and $v \in A_\sharp^\infty$ be two words, possibly drawn at random. We say u and v are h-equivalent, written $u \overset{h}{\equiv} v$, if they almost surely have the same h-contractions.*

Two words are h-equivalent if their contracted forms yield the same trajectories, as illustrated in Figure 2. Since contraction removes only passive letters, the words themselves give similar trajectories, but with pauses inserted at different moments, during which the trajectory is constant.

Property 7. *Let $h \in A_\sharp^*$ be a history, $q \in [0,1]$, and $u \in A_\sharp^\infty$. We have that*

$$c^h(u) \overset{h}{\equiv} u \quad \text{and} \quad e_q^h(u) \overset{h}{\equiv} u,$$

and therefore that

$$e_q^h\left(c^h(u)\right) \overset{h}{\equiv} u. \tag{1}$$

Additionally, if $u \in A_\sharp^*$, $v \in A_\sharp^*$ and $u \overset{h}{\equiv} v$, then

$$\mathcal{S} \circ h \circ u = \mathcal{S} \circ h \circ v. \tag{2}$$

Finally, under the same assumptions,

$$c^{h\cdot u} = c^{h\cdot v} \quad \textit{and} \quad e_q^{h\cdot u} = e_q^{h\cdot v}. \tag{3}$$

These results are straightforward, but will be used throughout the rest of this section to better analyse the effects of contraction and expansion. Note that the reciprocal of (2) is not true: two trajectories ending in the same state are not necessarily equivalent.

We now justify the above statement that expansion somewhat reconstructs contracted words.

Property 8. *Let $u \sim D_q^{\otimes\mathbb{N}}$ and $h \in A_\sharp^*$. We have that $e_q^h\left(c^h(u)\right) \sim D_q^{\otimes\mathbb{N}}$.*

Proof. Let $v = e_q^h\left(c^h(u)\right)$, $l \in \mathbb{N}$ and $a_{1\to l} \in A_\sharp^l$. We have that

$$\mathbf{P}\left\{v_{1\to l} = a_{1\to l}\right\} = \prod_{k=1}^{l} \mathbf{P}\left\{v_k = a_k \mid v_{1\to k-1} = a_{1\to k-1}\right\}.$$

Showing that, for all $k \in \mathbb{N}$,

$$\mathbf{P}\left\{v_k = a_k \mid v_{1\to k-1} = a_{1\to k-1}\right\} = D_q(a_k) \tag{4}$$

would yield that $v_{1\to l} \sim D_q^{\otimes l}$. This being true for all $l \in \mathbb{N}$, we would in turn have that

$$v \sim D_q^{\otimes\mathbb{N}}.$$

We now show (4) by differentiating the two cases where v_k is or is not in $\mathfrak{P}_{h\cdot a_{1\to k-1}}$:

$$\begin{aligned}
&\mathbf{P}\left\{v_k = a_k \mid v_{1\to k-1} = a_{1\to k-1}\right\} \\
&= \mathbf{P}\left\{v_k = a_k \;\middle|\; v_k \in \mathfrak{P}_{h\cdot a_{1\to k-1}} \bigwedge v_{1\to k-1} = a_{1\to k-1}\right\} \\
&\qquad \times \mathbf{P}\left\{v_k \in \mathfrak{P}_{h\cdot a_{1\to k-1}} \;\middle|\; v_{1\to k-1} = a_{1\to k-1}\right\} \\
&\quad + \mathbf{P}\left\{v_k = a_k \;\middle|\; v_k \in \mathfrak{A}_{h\cdot a_{1\to k-1}} \bigwedge v_{1\to k-1} = a_{1\to k-1}\right\} \\
&\qquad \times \mathbf{P}\left\{v_k \in \mathfrak{A}_{h\cdot a_{1\to k-1}} \;\middle|\; v_{1\to k-1} = a_{1\to k-1}\right\}.
\end{aligned}$$

Notice that v_k is in $\mathfrak{P}_{h\cdot a_{1\to k-1}}$ if and only if it was added during expansion. By definition, this occurs with probability $D_q\left(\mathfrak{P}_{h\cdot a_{1\to k-1}}\right)$, so we have that

$$\mathbf{P}\left\{v_k \in \mathfrak{P}_{h\cdot a_{1\to k-1}} \;\middle|\; v_{1\to k-1} = a_{1\to k-1}\right\} = D_q\left(\mathfrak{P}_{h\cdot a_{1\to k-1}}\right)$$

and

$$\mathbf{P}\left\{v_k \in \mathfrak{A}_{h\cdot a_{1\to k-1}} \;\middle|\; v_{1\to k-1} = a_{1\to k-1}\right\} = D_q\left(\mathfrak{A}_{h\cdot a_{1\to k-1}}\right).$$

Consider the case where v_k is passive. It was inserted during expansion, and its distribution is therefore $D_q|_{\mathfrak{P}_{h\cdot a_{1\to k-1}}}$:

$$\begin{aligned}
&\mathbf{P}\left\{v_k = a_k \;\middle|\; v_k \in \mathfrak{P}_{h\cdot a_{1\to k-1}} \bigwedge v_{1\to k-1} = a_{1\to k-1}\right\} \\
&= D_q\left(a_k \;\middle|\; \mathfrak{P}_{h\cdot a_{1\to k-1}}\right).
\end{aligned}$$

Similarly, if v_k is known to be active, then it was already in u and was not removed during contraction. Its distribution was therefore D_q conditioned to being active, i.e. $D_q|_{\mathfrak{A}_{h\cdot a_{1\to k-1}}}$:

$$\begin{aligned}
&\mathbf{P}\left\{v_k = a_k \;\middle|\; v_k \in \mathfrak{A}_{h\cdot a_{1\to k-1}} \bigwedge v_{1\to k-1} = a_{1\to k-1}\right\} \\
&= D_q\left(a_k \;\middle|\; \mathfrak{A}_{h\cdot a_{1\to k-1}}\right).
\end{aligned}$$

Combining all these results gives that

$$\begin{aligned}
&\mathbf{P}\left\{v_k = a_k \mid v_{1\to k-1} = a_{1\to k-1}\right\} \\
&\quad = \; D_q\left(a_k \mid \mathfrak{P}_{h\cdot a_{1\to k-1}}\right) \times D_q\left(\mathfrak{P}_{h\cdot a_{1\to k-1}}\right) \\
&\qquad + D_q\left(a_k \mid \mathfrak{A}_{h\cdot a_{1\to k-1}}\right) \times D_q\left(\mathfrak{A}_{h\cdot a_{1\to k-1}}\right) \\
&\quad = \; D_q\left(a_k\right).
\end{aligned}$$

This concludes the proof. □

Let $u \sim D_q^{\otimes\mathbb{N}}$, and $u_{1\to\sharp}$ be the same word truncated after the first appearance of the letter $\sharp$. Call G_q the distribution of $u_{1\to\sharp}$.

Property 9. *Let $u_{1\to\sharp} \sim G_q$ and $h \in A_\sharp^*$. We have that $e_q^h\left(c^h\left(u_{1\to\sharp}\right)\right) \sim G_q$.*

Proof. Let $u \sim D_q^{\otimes\mathbb{N}}$ such that $u_{1\to\sharp}$ is u truncated after the first $\sharp$.

By definition, $\sharp$ is always active, and is therefore neither removed when contracting nor inserted when expanding. As a result, $e_q^h\left(c^h\left(u_{1\to\sharp}\right)\right)$ is $e_q^h\left(c^h\left(u\right)\right)$ truncated after the first $\sharp$. Combining this and the fact that, according to Property 8, $e_q^h\left(c^h\left(u\right)\right) \sim D_q^{\otimes\mathbb{N}}$, we have that $e_q^h\left(c^h\left(u_{1\to\sharp}\right)\right) \sim G_q$. □

This property justifies the claim that expansion corresponds to reconstructing (in distribution) a word that has been contracted, as this is indeed the case when the original word is drawn according to G_q.

For $n \in \mathbb{N}$, consider a sequence of words $\left(u^m\right)_{m\in[\![1,n]\!]}$, independently distributed according to $G_{2^{-m}}$, and call $\mathcal{G}_n$ the distribution of

$$u^n \cdot u^{n-1} \cdot \ldots \cdot u^1.$$

We now define the $\mathcal{G}$-expansion of a word. The aim is once again to rebuild what a word "could have been" before it was contracted, supposing it was initially drawn according to $\mathcal{G}_n$ for some $n \in \mathbb{N}$.

Formally, given a word v finishing with its nth $\sharp$, we first split it into a sequence of words $v^n \cdot \ldots \cdot v^1$ such that each v^m contains exactly one $\sharp$, which is its last letter. Notice that this decomposition is unique. We then define the $\mathcal{G}$-expansion of v as

$$e_{\mathcal{G}}\left(v\right) = e_{2^{-n}}^{\epsilon}\left(v^n\right) \cdot \ldots \cdot e_{2^{-m}}^{v^n\cdot\ldots\cdot v^{m+1}}\left(v^m\right) \cdot \ldots \cdot e_{\frac{1}{2}}^{v^n\cdot\ldots\cdot v^2}\left(v^1\right),$$

or ϵ, if $n = 0$.

Property 10. *Let $u \sim \mathcal{G}_n$. We have that*

$$e_{\mathcal{G}}\left(c^{\epsilon}\left(u\right)\right) \sim \mathcal{G}_n$$

and

$$e_{\mathcal{G}}\left(c^{\epsilon}\left(u\right)\right) \overset{\epsilon}{\equiv} u.$$

Proof. Notice that, by definition of contraction,

$$c^h\left(v \cdot w\right) = c^h\left(v\right) \cdot c^{h\cdot v}\left(w\right) \tag{5}$$

for any finite words v, w and h. Let $u^n \cdot u^{n-1} \cdot \ldots \cdot u^1$ be the unique decomposition of u such that each u^m ends with its unique sharp. We have that

$$e_{\mathcal{G}}\left(c^{\epsilon}\left(u\right)\right) = e_{\mathcal{G}}\left(c^{\epsilon}\left(u^n\right) \cdot \ldots \cdot c^{u^n\cdot\ldots\cdot u^2}\left(u^1\right)\right).$$

Since each $c^{u^n\cdot\ldots\cdot u^{m+1}}\left(u^m\right)$ must also finish with its unique $\sharp$, the definition of $e_{\mathcal{G}}$ gives that this is equal to

$$\begin{aligned}
&e^{\epsilon}_{2^{-n}}\left(c^{\epsilon}\left(u^n\right)\right)\cdot\ldots\cdot e^{c^{\epsilon}(u^n)\cdot\ldots\cdot c^{u^n\cdot\ldots\cdot u^3}\left(u^2\right)}_{\frac{1}{2}}\left(c^{u^n\cdot\ldots\cdot u^2}\left(u^1\right)\right)\\
&=e^{\epsilon}_{2^{-n}}\left(c^{\epsilon}\left(u^n\right)\right)\cdot\ldots\cdot e^{c^{\epsilon}\left(u^n\cdot\ldots\cdot u^2\right)}_{\frac{1}{2}}\left(c^{u^n\cdot\ldots\cdot u^2}\left(u^1\right)\right)\\
&=e^{\epsilon}_{2^{-n}}\left(c^{\epsilon}\left(u^n\right)\right)\cdot\ldots\cdot e^{u^n\cdot\ldots\cdot u^2}_{\frac{1}{2}}\left(c^{u^n\cdot\ldots\cdot u^2}\left(u^1\right)\right).
\end{aligned}$$

The last line is a consequence of (3), since

$$\forall m\in[\![1,n]\!],\ c^{\epsilon}\left(u^n\cdot\ldots\cdot u^m\right)\overset{\epsilon}{\equiv}u^n\cdot\ldots\cdot u^m.$$

Let

$$v^m=e^{u^n\cdot\ldots\cdot u^{m+1}}_{2^{-m}}\left(c^{u^n\cdot\ldots\cdot u^{m+1}}\left(u^m\right)\right)$$

for all $m\in[\![1,n]\!]$, such that

$$e_{\mathcal{G}}\left(c^{\epsilon}\left(u\right)\right)=v^n\cdot\ldots\cdot v^m\cdot\ldots\cdot v^1.$$

By definition, every u^m is distributed according to $G_{2^{-m}}$. Property 9 therefore gives us that every v^m is also distributed according to $G_{2^{-m}}$, which in turn implies that v is distributed according to $\mathcal{G}_n$.

Using (1), we also have that, for all $m\in[\![1,n]\!]$, v^m and u^m are $\left(u^n\cdot\ldots\cdot u^{m+1}\right)$-equivalent, that is to say

$$c^{u^n\cdot\ldots\cdot u^{m+1}}\left(v^m\right)=c^{u^n\cdot\ldots\cdot u^{m+1}}\left(u^m\right).$$

By concatenating and merging these using (5), we obtain that

$$c^{\epsilon}\left(v^n\cdot\ldots\cdot v^1\right)=c^{\epsilon}\left(u^n\cdot\ldots\cdot u^1\right),$$

i.e. $e_{\mathcal{G}}\left(c^{\epsilon}\left(u\right)\right)\overset{\epsilon}{\equiv}u$. □

Consider a sequence of words $\left(u^m\right)_{m\in\mathbb{N}}$, independently distributed such that for all $m\in\mathbb{N}$, $u^m\sim G_{2^{-m}}$. Define the sequence w^n recursively such that $w^0=\epsilon$ and

$$w^{n+1}=c^{\epsilon}\left(u^{n+1}\cdot e_{\mathcal{G}}\left(w^n\right)\right).$$

This is the basis for our CFTP algorithm with oracle skipping: if w^n is not a coupling word, then compute w^{n+1}, repeating the operation until a coupling word is found.

Property 11. *Using the above notation, we have that, for all $m<n$, $\mathcal{S}\circ w^n\subseteq\mathcal{S}\circ w^m$. In other words, the trajectories obtained at each iteration are embedded in one another.*

Proof. It is enough to show that, for all $n\in\mathbb{N}$,

$$\mathcal{S}\circ w^{n+1}\subseteq\mathcal{S}\circ w^n.$$

Since w^{n+1} and $u^{n+1}\cdot e_{\mathcal{G}}\left(w^n\right)$ are ϵ-equivalent, (2) gives us that

$$\begin{aligned}
\mathcal{S}\circ w^{n+1}&=\mathcal{S}\circ u^{n+1}\circ e_{\mathcal{G}}\left(w^n\right)\\
&\subseteq\mathcal{S}\circ e_{\mathcal{G}}\left(w^n\right),
\end{aligned}$$

the inclusion being a direct consequence of the fact that $\mathcal{S}\circ u^{n+1}\subseteq\mathcal{S}$. Since w^n is its own ϵ-contraction, Property 10 gives that w^n and $e_{\mathcal{G}}\left(w^n\right)$ are also ϵ-equivalent, and (2) yields that

$$\mathcal{S}\circ e_{\mathcal{G}}\left(w^n\right)=\mathcal{S}\circ w^n,$$

which concludes the proof. □

Let

$$N = \inf\left\{n \in \mathbb{N} \;\middle|\; |\mathcal{S} \circ w^n| = 1\right\}$$

be the first iteration at which a coupling word is found.

Theorem 2. *Using the above notation, we have that, if there exists a coupling word for the bounding chain of $\mathcal{A}$, then*

$$\mathbf{P}\left\{N < +\infty\right\} = 1,$$

and

$$\mathbf{E}\left[N\right] < +\infty,$$

i.e. the algorithm almost surely terminates, and does so after a finite expected number of iterations.

Furthermore, The unique element of $\mathcal{S} \circ w^N$ is distributed according to the stationary distribution π of $\mathcal{A}$.

Note that, due to Property 11, for every $n \geq N$, $\mathcal{S} \circ w^n$ is a singleton that contains the same state, distributed according to π.

Proof. The proof is fundamentally the same as that of Theorem 1.

For the first part of the theorem, let $u \in A^*$ be a coupling word for the bounding chain of $\mathcal{A}$, and $P_u = D^{\otimes|u|}(u)$ be the probability of drawing a word with prefix u. At each iteration, the probability of u^k having u as a prefix is the probability of there being no $\sharp$ in the first $|u|$ letters, i.e. having $\left|u^k\right| > |u|$, times the probability of the prefix being u knowing there are no $\sharp$, i.e. P_u. We therefore have that

$$\begin{aligned}
\mathbf{P}\left\{u \text{ prefix of } u^k\right\} &= \mathbf{P}\left\{\left|u^k\right| > |u|\right\} P_u \\
&\geq \mathbf{P}\left\{\left|u^1\right| > |u|\right\} P_u \\
&\geq \left(\frac{1}{2}\right)^{|u|} P_u \\
&> 0.
\end{aligned}$$

This implies we will almost surely draw a word u^k that couples, at which point w^k will also couple and the algorithm will stop. Furthermore, the expected number of iterations is finite, as it is upper-bounded by

$$\frac{2^{|u|}}{P_u}.$$

We now show that the output state is distributed according to π. Call x_{out} the unique element of $\mathcal{S} \circ w^N$. If we can show that, for all $\varepsilon > 0$ and all $x \in \mathcal{S}$,

$$\left|\mathbf{P}\left\{x_{\text{out}} = x\right\} - \pi(x)\right| \leq \varepsilon, \tag{6}$$

then we are finished.

Decompose $e_{\mathcal{G}}\left(w^N\right)$ as

$$e_{\mathcal{G}}\left(w^N\right) = u^N \cdot \sharp \cdot u^{N-1} \cdot \sharp \cdot \ldots \cdot \sharp \cdot u^1 \cdot \sharp,$$

with $u^m \in A^*$ for all m. Let $u_{-\infty\to 0}^{\infty} \sim D^{\otimes\mathbb{Z}^-}$, and

$$\widetilde{w}_{-\infty\to 0} = u_{-\infty\to 0}^{\infty} \cdot u^N \cdot u^{N-1} \cdot \ldots \cdot u^1.$$

We begin by showing that $\widetilde{w}_{-\infty\to 0} \sim D^{\otimes\mathbb{Z}^-}$.

Consider a word $v_{-\infty\to 0} \sim D^{\otimes\mathbb{Z}^-}$ and a family of random, independent instances $(k_m)_{m\in[\![1,N]\!]}$ such that, for each m, k_m is distributed according to a geometric distribution of parameter 2^{-m}. Let $l_m = \sum_{i=1}^{m} k_i$ for all $m \in [\![0, N]\!]$.

Since $u^m \cdot \sharp \sim G_{2^{-m}}$ for all m, the lengths of the u^m are equal to the position of the first $\sharp$ in a sequence of letters i.i.d. according to $D_{2^{-m}}^{\otimes \mathbb{N}}$, minus one. This is exactly the geometric distribution of parameter 2^{-m}. In particular, we have that

$$u^m \sim v_{-l_m+1 \to -l_{m-1}},$$

and therefore

$$\begin{aligned}
\widetilde{w}_{-\infty \to 0} &= u^{\infty}_{-\infty \to 0} \cdot u^N \cdot \ldots \cdot u^1 \\
&\sim v_{-\infty \to -l_N} \cdot v_{-l_N+1 \to -l_{N-1}} \cdot \ldots \cdot v_{-l_1+1 \to -l_0} \\
&= v_{-\infty \to 0} \\
&\sim D^{\otimes -\mathbb{N}}.
\end{aligned}$$

Let $l = \left|u^N \cdot u^{N-1} \cdot \ldots \cdot u^1\right|$, such that

$$\widetilde{w}_{-l+1 \to 0} = u^N \cdot u^{N-1} \cdot \ldots \cdot u^1.$$

Notice that

$$\begin{aligned}
\{x_{\text{out}}\} &= \mathcal{S} \circ w^N \\
&= \mathcal{S} \circ e_{\mathcal{G}}\left(w^N\right) \\
&= \mathcal{S} \circ \widetilde{w}_{-l+1 \to 0}.
\end{aligned}$$

We now show that if, for some $k \in \mathbb{N}$, $\widetilde{w}_{-k \to 0}$ is a coupling word, then $\mathcal{S} \circ \widetilde{w}_{-k \to 0} = \{x_{\text{out}}\}$.
If $k \geq l$, then

$$\begin{aligned}
\{x_{\text{out}}\} &= \mathcal{S} \circ \widetilde{w}_{-l+1 \to 0} \\
&= \mathcal{S} \circ \widetilde{w}_{-l+1 \to -k-1} \circ \widetilde{w}_{-k \to 0} \\
&\subseteq \mathcal{S} \circ \widetilde{w}_{-k \to 0},
\end{aligned}$$

and if $k < l$, then

$$\begin{aligned}
\mathcal{S} \circ \widetilde{w}_{-k \to 0} &= \mathcal{S} \circ \widetilde{w}_{-k \to -l} \circ \widetilde{w}_{-l+1 \to 0} \\
&\subseteq \mathcal{S} \circ \widetilde{w}_{-l+1 \to 0} \\
&= \{x_{\text{out}}\}.
\end{aligned}$$

In both cases, if $\widetilde{w}_{-k \to 0}$ is a coupling word, then $\mathcal{S} \circ \widetilde{w}_{-k \to 0}$ is a singleton, and therefore necessarily equal to $\{x_{\text{out}}\}$.

We now show (6). Since the bounding chain couple a.s., there exists $t_\varepsilon \in \mathbb{N}$ such that

$$\mathbf{P}\left\{\left|\mathcal{S} \circ \widetilde{w}_0 \circ \widetilde{w}_{-1} \circ \ldots \circ \widetilde{w}_{-t_\varepsilon}\right| = 1\right\} \geq 1 - \varepsilon,$$

and since $\widetilde{w}_0 \cdot \widetilde{w}_{-1} \cdot \ldots \cdot \widetilde{w}_{-t_\varepsilon}$ has the same distribution as $\widetilde{w}_{-t_\varepsilon \to 0}$, we have that

$$\mathbf{P}\left\{\left|\mathcal{S} \circ \widetilde{w}_{-t_\varepsilon \to 0}\right| = 1\right\} \geq 1 - \varepsilon.$$

Let $Y = y \cdot \widetilde{w}_{-t_\varepsilon \to 0}$, with $y \sim \pi$. Notice that $Y \in \mathcal{S} \circ \widetilde{w}_{-t_\varepsilon \to 0}$, and that the letters in $\widetilde{w}_{-t_\varepsilon \to 0}$ are i.i.d. according to D, such that $Y \sim \pi$.

If $\left|\mathcal{S} \circ \widetilde{w}_{-t_\varepsilon \to 0}\right| = 1$, then $\mathcal{S} \circ \widetilde{w}_{-t_\varepsilon \to 0} = \{x_{\text{out}}\}$, i.e. $Y = x_{\text{out}}$. We therefore have that

$$\mathbf{P}\left\{x_{\text{out}} \neq Y\right\} \leq \mathbf{P}\left\{\left|\mathcal{S} \circ \widetilde{w}_{-t_\varepsilon \to 0}\right| > 1\right\} \leq \varepsilon,$$

and thus

$$\forall x \in S,\ \left|\mathbf{P}\left\{x_{\text{out}} = x\right\} - \mathbf{P}\left\{Y = x\right\}\right| \leq \varepsilon.$$

This is precisely (6). □

The aim of skipping is to keep only the contracted words, and avoid the expanded ones. Indeed, the extended words correspond to what the initial CFTP algorithm stores. Notice that when computing w^{n+1}, w^n is expanded and then contracted. In practice, it is possible to combine these operations so as to only insert letters that are now active, rather then all "potentially skipped" letters. Contracting then only removes letters that were active at the previous iteration. This ensures that the size of the word (and therefore the time spent reading it) remains that of the contracted form.

We implement this algorithm using First In First Out queues to represent words. It is given in Algorithm 3. Note that there is often an *ad hoc* means of computing the different sets of active letters dynamically, rather than recomputing them at each iteration.

We now give an important result for computing some upper-bounds on the computation time of our algorithm.

Property 12. *Using the previous notation, call $\tau_{\mathcal{O}}^f$ the coupling time of the bounding chain of $\mathcal{A}$, and $\tau_{\mathcal{O}}^b = \left|w^N\right|$ coupling time of the corresponding CFTP algorithm with oracle skipping. If*

$$D^{\otimes k+1}\left\{a_{k+1} \in \mathfrak{P}_{a_{1\to k}}\right\} \tag{7}$$

is increasing in k, i.e. passive letters become more likely as time passes, then

$$\mathbf{E}\left[\tau_{\mathcal{O}}^b\right] \leq 2 \cdot \left(\mathbf{E}\left[N\right] + \mathbf{E}\left[\tau_{\mathcal{O}}^f\right]\right).$$

The proof is the same as for the initial CFTP algorithm, with the following two exceptions:

- Since the algorithm computes transitions beyond the coupling of the bounding chain, it is important to make sure the proportion of active events after coupling does not exceed the proportion during coupling. This is ensured by condition (7).
- For the CFTP algorithm with oracle skipping, we introduce a delimiter $\sharp$ in our coupling word, which is not present in the initial bounding chain. This delimiter is present N times in the final coupling word, hence the $\mathbf{E}\left[N\right]$ term to account for this.

Notice that, by construction, the number of times the algorithm goes back in time is the same as with normal CFTP. Since $\sharp$ is added exactly once every time the algorithm does so, we have that $\mathbf{E}\left[N\right]$ is equal to $\mathbf{E}\left[\log_2\left(\tau^b\right)\right]$, where τ^b is the backward coupling time of the algorithm without skipping. The overall complexity of the algorithm is therefore in $O\left(\mathbf{E}\left[\tau_{\mathcal{O}}^f\right]\right)$, so long as this does no better than $O\left(\mathbf{E}\left[\log_2\left(\tau^b\right)\right]\right)$.

This serves as a motivation to study the average forward coupling time of the Markov automaton with oracle skipping, which can serve as a means of estimating the average coupling time of the CFTP algorithm with either oracle or incremental skipping.

4 Independent Sets

Let $G = (V, E)$ be a simple undirected graph. A subset I of V is called an independent set if no two vertices in I are connected by an edge, i.e. if

$$\forall x, y \in I, (x, y) \notin E.$$

Let $\mathcal{I}$ be the set of independent sets of G and, for any vertex $v \in V$, denote $N\left(v\right)$ the set of neighbors of v, that is to say the $w \in V$ such that $(v, w) \in E$.

We study the performance of the CFTP algorithm with oracle skipping when sampling independent sets according to the distribution

$$P_\lambda\left(I\right) = \frac{\lambda^{|I|}}{Z_\lambda}, \lambda \in \mathbb{R},$$

focusing on the case where λ is very large. Due to Property 12, we restrict our analysis to the complexity for the forward coupling.

Algorithm 3 CFTP with Oracle Skipping

function ORACLE-CFTP($\mathcal{A} = (\mathcal{S}, A, D, \cdot)$)
  $n \leftarrow 0$, $w \leftarrow []$ — *[] is the empty queue*
  repeat
    INCREMENT(n)
    $(B, w) \leftarrow$ DOUBLE-HISTORY($\mathcal{A}, w, n$)
  until $|B| = 1$
  return ELEMENTOF(B)
end function

function DOUBLE-HISTORY($\mathcal{A}, w, n$)
  $v \leftarrow w$, $m \leftarrow n$
  $(B^+, \mathfrak{A}^+, u) \leftarrow$ G-WORD($\mathcal{A}, 2^{-m}$) — *Compute the contracted u^n*
  $B^- \leftarrow \mathcal{S}$ — *B^+ and B^- are the new and old bounding chains*
  $\mathfrak{A}^- \leftarrow \mathfrak{A}(\mathcal{S})$
  DECREMENT(m)
  while NOTEMPTY(v) **do** — *Expand and contract w^{n-1}*
    $\mathfrak{A} \leftarrow \mathfrak{A}^+ \cup \mathfrak{A}^-$ — ***Step 1**: Expanding*
    $a \leftarrow$ DRAW($D_{2^{-m}}|_{\mathfrak{A}}$) — *Try to expend by one letter...*
    if $a \in \mathfrak{A}^-$ **then** — *... but keep it only if it is passive*
      $a \leftarrow$ POP(v) — *Otherwise, take the next letter in v...*
      $(B^-, \mathfrak{A}^-) \leftarrow (B^-, \mathfrak{A}^-) \circ a$ — *... and update the previous chain*
    end if
    if $a \in \mathfrak{A}^+$ **then** — ***Step 2**: contracting*
      PUSH(a, u) — *Only keep active letters...*
      $(B^+, \mathfrak{A}^+) \leftarrow (B^+, \mathfrak{A}^+) \circ a$ — *... and update the new chain*
      if $a = \sharp$ **then**
        DECREMENT(m) — *Probability of seeing $\sharp$ has changed*
      end if
    end if
  end while
  return (B^+, u)
end function

function G-WORD($\mathcal{A}, p$)
  $u \leftarrow []$ — *[] is the empty queue*
  $B \leftarrow \mathcal{S}$ — *Bounding chain*
  $\mathfrak{A} \leftarrow \mathfrak{A}(\mathcal{S})$
  repeat
    $a \leftarrow$ DRAW($D_q|_{\mathfrak{A}}$)
    PUSH(a, u)
    $(B, \mathfrak{A}) \leftarrow (B, \mathfrak{A}) \circ a$
  until $a = \sharp$
  return $(B, \mathfrak{A}, u)$
end function

4.1 Sampling algorithms

We compare the coupling time our sampling algorithm with oracle skipping with two other approaches described in [7]: Gibbs sampling and the Dyer-Greenhill chain [2].

4.1.1 Gibbs sampling

Let us first define a Gibbs sampler for P_λ. At each iteration, independently draw a vertex v uniformly at random and u uniformly over $[0, 1]$.

- If $u > \frac{\lambda}{\lambda+1}$, then remove v from I if $v \in I$, otherwise do nothing.
- If $0 \leq u \leq \frac{\lambda}{\lambda+1}$, then add v to I if $N(v) \cap I = \emptyset$, otherwise do nothing.

This dynamic allows us to use Monte Carlo and CFTP methods to generate independent sets according to P_λ. The CFTP approach can be greatly improved by using the following bounding chain for the Glauber dynamic defined in [7].

Consider a family of independent sets $A \subseteq \mathcal{I}$. Set

$$B = \cap_{I \in A} I, \qquad D = (\cup_{I \in A} I) \setminus B$$

and

$$C = \cap_{I \in A}(V \setminus I) = V - B - D.$$

We have that

$$A \subseteq \{I \in \mathcal{I} | B \subseteq I \subseteq B \cup D\} = \langle B, D \rangle.$$

In other words, B is the set of vertices common to every independent set in A, C is the set of vertices that are in none of the independent sets of A, and D is the set of vertices that are in some but not all of the independent sets of A. The couples $(\langle B_i, D_i \rangle)_{i \in \mathbb{N}}$ define a bounding chain for the Glauber dynamic $(A_i)_{i \in \mathbb{N}}$.

The Gibbs sampler for the bounding chain is defined as follows: at each iteration, independently draw a vertex v uniformly at random and u uniformly over $[0, 1]$. Suppose the initial state is $\langle B, D \rangle$, and write $B + v$ for $B \cup \{v\}$ and $B - v$ for $B \setminus \{v\}$; the arrival state $\langle B', D' \rangle$ is constructed as follows:

- If $u > \frac{\lambda}{\lambda+1}$, then $B' = B - v$, $D' = D - v$.
- If $0 \leq u \leq \frac{\lambda}{\lambda+1}$, then:
  - if $N(v) \subseteq C$, then $B' = B + v$ and $D' = D - v$,
  - if $N(v) \cap B = \emptyset$ but $N(v) \cap D \neq \emptyset$, then $D' = D + v$ (v was necessarily in $C \cup D$),
  - otherwise do nothing (v was necessarily in C).

4.1.2 The Dyer-Greenhill scheme

The coupling time of the above bounding chain can be reduced through the Dyer-Greenhill scheme. The main idea is to enable two elements in the independent set to swap positions. Given $p_s \in [0, 1]$, if, in the Gibbs sampler, an attempt to add v to the independent set I fails due to the presence of a *unique* neighbour u already in I, then with probability p_s, the independent set becomes $I + v - u$. A bounding chain can easily be defined for this new scheme.

4.1.3 Oracle skipping scheme

Now consider oracle skipping for the bounding chain of the Gibbs sampler. For each vertex v, we have two events: adding v to I, denoted a_v, and removing v from I, denoted r_v. The active events are:

- the r_v for which $v \notin C$,
- the a_v for which $v \in C$ and $N(v) \cap B = \emptyset$,
- the a_v for which $v \in D$ and $N(v) \subseteq C$.

Let V_r and V_a be the set of vertices for which removal and addition are respectively active in $\langle B, D\rangle$. For the Gibbs sampler, events are drawn according to the conditional distribution by picking an event uniformly at random in V_z, where $z = a$ with probability $\lambda |V_a| / (\lambda |V_a| + |V_r|)$, and $z = r$ otherwise.

For a vertex $v \in V$, the fact that a_v and r_v are active is only modified when v, or one of its neighbours, is modified. It is therefore possible to locally update the conditional distribution at each iteration by simply updating the "activeness" of events for the modified vertex and its neighbours. This justifies using oracle skipping rather than incremental skipping in this context.

Note that those three samplers can be adapted to the case of weighted vertices and product-form stationary processes of the form

$$P_\Lambda(I) = \frac{1}{Z_\Lambda} \prod_{v \in I} \lambda(v),$$

where $\Lambda = (\lambda_v)_{v \in V}$ is a weight-vector of the vertices. For the Gibbs sampler, λ is replaced by the $\lambda(v)$ of the selected vertex. The other samplers are modified accordingly.

4.2 Star graph

In this paragraph, we study the graph

$$G_n = \left([\![0, n]\!], \{(0, i), i \in [\![1, n]\!]\}\right),$$

called *star graph*. We focus mainly on the performance of the oracle skipping scheme for large values of λ, such as when $\lambda \gg n$. The independents of this graph are

$$\mathcal{I} = \{\{0\}\} \cup \mathcal{P}([\![1, n]\!]).$$

First, we consider the coupling time τ of the Glauber dynamic of the bounding chains without skipping, both in the case of the Gibbs sampler and of the Dyer-Greenhill sampler.

Since at most one vertex is removed from D at each iteration, and the algorithm finishes when $D = \emptyset$, this coupling time is lower bounded by the hitting time of

$$\langle B, \{0\}\rangle \cup \langle B, D\rangle,\ 0 \notin D.$$

Furthermore, since no vertex can be added to B so long as D contains both 0 and an element in $[\![1, n]\!]$, $B = \emptyset$ until one of those states is reached.

In the case of the Gibbs sampler, if $\lambda > 1$, the expected hitting time of $\langle \emptyset, \{0\}\rangle$ is $O(\lambda^n)$. Furthermore, before reaching this state, the probability of removing 0 from D is exactly $\frac{1}{(\lambda+1)(n+1)}$ at each time step. For n large enough, this gives

$$\mathbf{E}[\tau] \geq (n+1)(\lambda+1).$$

For the Dyer-Greenhill sampler, the coupling time τ^{DG} is greatly reduced: the expected hitting time of $\langle B, D\rangle$, $0 \notin D$ is constant:

$$E = \frac{\lambda+1}{\lambda} \frac{n+1}{n} \frac{1}{p_s}.$$

This is due to the fact that the first attempt to swap a vertex other than 0 will immediately remove 0 from D, since it is the only neighbour of the selected vertex.

On the other hand, for the bounding chain to couple, every vertex must be selected at least once for addition or removal. As at each step, the modified vertex is chosen uniformly and independently at random, this gives that

$$\mathbf{E}\left[\tau^{DG}\right] \geq n \ln n + O(1).$$

Now, let us consider the (forward) coupling time $\tau^{\mathcal{O}}$ of the coupling chain with oracle skipping. The coupling time is at most the hitting time of $\langle B, \emptyset \rangle$. We have two main steps to consider:

1. The hitting time of a state $\langle B, D \rangle$ where $0 \notin D$;
2. The hitting time of $\langle B, \emptyset \rangle$.

Let us first focus on the hitting time of $\langle B, D \rangle$ where $0 \notin D$. Consider the following birth-and-death process on $[\![0, n]\!]$, where state i represents the cardinal of C, until 0 is added to C. In state i, the active events are the addition of vertices in C and the removal of vertices in D. As a consequence, the probabilities $p_{i,i+1}$ and $p_{i+1,i}$ to go respectively from state i to state $i+1$ and from $i+1$ to i are

$$p_{i,i+1} = \frac{n-i}{n-i+i\lambda} \text{ and } p_{i+1,i} = \frac{(i+1)\lambda}{n-i-1+(i+1)\lambda}. \tag{8}$$

As we assumed $\lambda \geq n$, computations show that the stationary distribution π of this birth-and-death process satisfies, for all $i \in [\![0, n]\!]$,

$$\pi(i) = \left(\binom{n-1}{i-1} \lambda^{-(i-1)} + \binom{n-1}{i} \lambda^{-i} \right) \pi(0).$$

so $\pi(0) \geq \frac{1}{2}\left(1 + \frac{1}{\lambda}\right)^{-n}$.

The bounding chain can be bounded by the following process: when in state 0 only, vertex 0 can be removed with probability $1/(n+1)$ (all events are active for removal, none for addition). Then the expected time τ_1 for reaching a state $\langle B, D \rangle$ where $0 \notin D$ is (when $\lambda \geq n$)

$$\mathbf{E}\left[\tau_1\right] = \frac{(n+1)}{\pi(0)} \leq 2e(n+1).$$

For the second step, consider the birth-and-death process on $[\![0, n]\!]$ where state i represents the $\langle B, D \rangle$ such that $|B| = n - i$ and $0 \notin D$. For $i > 0$, i vertices are active for addition and at least $n - i$ are active for removal. The transitions probabilities are exactly the probabilities $p_{i,j}$ defined in Eq. (8).

Simple computations show that the hitting time τ_2 of state 0 from state n satisfies

$$\mathbf{E}\left[\tau_2\right] = \left(\frac{1+\lambda}{\lambda}\right)^n + n \leq n + e^{n/\lambda} = n + O(1).$$

Finally, note that in state n, vertex 0 is active for addition, and in case this event is generated (which happens with probability $\frac{1}{n+1}$), we have to take into account the return time from the first step ($0 \in D$) to the second step ($0 \notin D$). By the Markov inequality, the probability that state n is visited again before state 0 is at most $\frac{\pi(n)}{\pi(0)} = \lambda^{-n}$.

As a consequence, the expected coupling time satisfies

$$\mathbf{E}\left[\tau^{\emptyset}\right] \leq \mathbf{E}\left[\tau_1\right] + \mathbf{E}\left[\tau_2\right] + O(1) \leq (2e+1)n + O(1).$$

Notice that the coupling time does not depend on λ and is linear in n. It therefore does better then the other samplers presented above.

4.3 Numerical experiments

We now do an experimental comparison of the three samplers described in Section 4.1 for two models: the star graph, that has been precisely analysed in Paragraph 4.2, and the Barabási-Albert model [1].

Star graph We performed experiments for a star graph with 100 vertices and for different values of λ. For each value of λ and each sampler, 1000 experiments have been performed, and the average number of transitions computed is depicted in Figures 3.

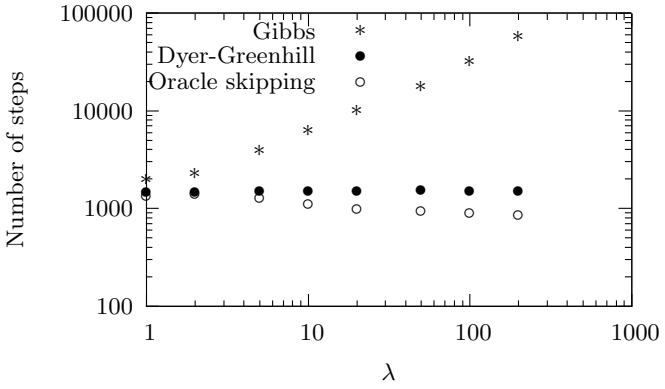


Figure 3: Number of events generated by CFTP algortihms for the star graph with 100 vertices for different values of λ.

The first remark is that both Dyer-Greenhill and oracle skipping samplers outperform the Gibbs sampler. Second, the Dyer-Greenhill sampler seams insensitive to the value of λ, which conforms to the bound $n \ln n + O(1)$ given in Section 4.2. Finally, the oracle skipping scheme is always the most efficient algorithm. It is noticeable that the number of event generated decreases with λ. This can be explained the following way: large independent sets are favored when λ grows. Then, after reaching the independent set $[\![1, n]\!]$, whose probability grows with λ, the probability that the next event is active is less than $1/(1+\lambda)$. As a consequence, many events are skipped.

The difference in behavior between the Dyer-Greenhill and oracle skipping samplers is more obvious with the star graph with 1000 vertices, as depicted in Figure 4 (100 experiments are run for each value of λ).

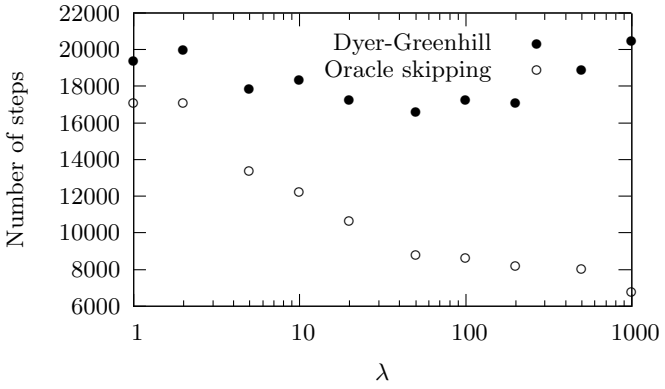


Figure 4: Number of events generated by CFTP algortihms for the star graph with 1000 vertices for different values of λ.

Barabási-Albert model We now generate a random graph with preferential attachment. Start from a clique with 5 vertices and at each step add one new vertex v and two edges (v, w_1) and (v, w_2), where w_1 and w_2 are chosen at random with probability proportional to their degree. Figure 5 compares the average number of events generated for 100 experiments with the three samplers, for graphs with 100 vertices.

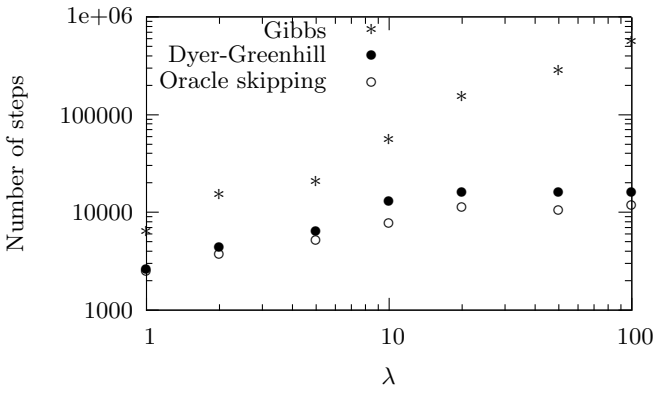


Figure 5: Number of events generated by CFTP algortihms for the Barabasi-Albert model with 100 vertices for different values of λ.

Similarly to the star graph, Dyer-Greenhill and oracle skipping samplers outperform the Gibbs sampler, and the oracle skipping sample is sensitively better than the Dyer-Greenhill one. For large values, those two samplers are not sensitive to λ (or slightly improve when λ grows).

5 Conclusions

The main contribution of the paper is Algorithm 3, that speeds up the Markovian dynamics in the CFTP scheme for exact sampling from the stationary distribution of a Markov chain. We illustrated it here on the problem of randomly generating independent sets, but its applicability is much broader, within the context of the random generation of combinatorial objects using Glauber dynamics, or elsewhere. The application to the simulation of queueing networks is an ongoing research.

References

[1] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Rev. Modern Phys.*, 74(1):47–97, 2002.

[2] M. Dyer and C. Greenhill. On Markov chains for independent sets. *J. Algorithms*, 35(1):17–49, 2000.

[3] A. El Gamal and Y.-H. Kim. *Network information theory.* Cambridge University Press, Cambridge, 2011.

[4] D. Galvin and P. Tetali. Slow mixing of glauber dynamics for the hard-core model on regular bipartite graphs. *Random Struct. Algorithms*, 28(4):427–443, July 2006.

[5] J. Ghaderi and R. Srikant. On the design of efficient CSMA algorithms for wireless networks. In *Proceedings of the 49th IEEE Conference on Decision and Control, CDC 2010, December 15-17, 2010, Atlanta, Georgia, USA*, pages 954–959. IEEE, 2010.

[6] O. Häggström. *Finite Markov chains and algorithmic applications*, volume 52 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 2002.

[7] M. Huber. Perfect sampling using bounding chains. *Ann. Appl. Probab.*, 14(2):734–753, 2004.

[8] L. Jiang, M. Leconte, J. Ni, R. Srikant, and J. C. Walrand. Fast mixing of parallel glauber dynamics and low-delay CSMA scheduling. *IEEE Transactions on Information Theory*, 58(10):6541–6555, 2012.

[9] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, Providence, RI, 2009. With a chapter by James G. Propp and David B. Wilson.

[10] F. Martinelli. Lectures on Glauber dynamics for discrete spin models. In *Lectures on probability theory and statistics (Saint-Flour, 1997)*, volume 1717 of *Lecture Notes in Math.*, pages 93–191. Springer, Berlin, 1999.

[11] F. Pin, A. Bušić, and B. Gaujal. Acceleration of perfect sampling by skipping events. In *Proceedings of the 5th International Conference on Performance Evaluation Methodologies and Tools (Valuetools)*, 2011.

[12] J. G. Propp and D. B. Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. *Random Structures & Algorithms*, 9(1-2):223–252, 1996.

[13] S. Sanghavi, D. Shah, and A. S. Willsky. Message passing for maximum weight independent set. *IEEE Trans. Inf. Theor.*, 55(11):4822–4834, Nov. 2009.

[14] D. Shah and J. Shin. Randomized scheduling algorithm for queueing networks. *Ann. Appl. Probab.*, 22(1):128–171, 2012.

[15] V. G. Subramanian and M. Alanyali. Delay performance of CSMA in networks with bounded degree conflict graphs. In A. Kuleshov, V. Blinovsky, and A. Ephremides, editors, *2011 IEEE International Symposium on Information Theory Proceedings, ISIT 2011, St. Petersburg, Russia, July 31 - August 5, 2011*, pages 2373–2377. IEEE, 2011.

[16] L. Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 453–461, New York, NY, USA, 2001. ACM.